IN THE CLAIMS

Claims 1-43 were pending at the time of the action.

Claims 1, 5, 6, 7, 10, 20, 25, 31, 35, 39, and 42 are amended in this communication.

Claim 4 is cancelled.

No claims are added.

1. **(Currently amended)** A method comprising:

receiving a request to upgrade a current trusted core of an operating system to a new trusted core, wherein the current trusted core is installed on a computing device; and

allowing the new trusted core to access application data previously securely stored by the current trusted core only if it can be verified that the new trusted core can be trusted and further only if the new trusted core has a strictly increasing version number relative to the current trusted core and is signed by a certification authority that also signed the current trusted core.

2. **(Original)** A method as recited in claim 1, wherein the allowing comprises allowing the new trusted core to access application data previously securely stored by the current trusted core only if it can be verified that the new trusted core is the new trusted core expected by the current trusted core.

3. **(Original)** A method as recited in claim 1, wherein the allowing comprises the current trusted core:

identifying a digest of the new trusted core; and

having a key securely generated so that it can only be retrieved by the new trusted core based on the digest of the new trusted core, wherein the key was previously used by the current trusted core to securely store data for applications executing on the same computing device as the current trusted core.

4. **(Cancelled).**

5. **(Currently amended)** A method as recited in claim [[4]] 3, further comprising:

checking, by the current trusted core, whether the new trusted core satisfies one or more conditions; and

having the key securely stored only if the new trusted core satisfies the one or more conditions.

6. **(Currently amended)** A method as recited in claim [[4]] 1, further comprising:

~~generating a temporary key;~~

~~encrypting, using the temporary key, a subset of the data securely stored for applications executing on the computing device; and~~

~~having the temporary key securely stored so that it can only be retrieved by the new trusted core, but not having the key securely stored so that it can be retrieved by the new trusted core.~~

storing trusted application data utilizing a gatekeeper storage key and a hive key, wherein the allowing comprises allowing the new trusted core to access the gatekeeper storage key.

7. **(Currently amended)** A method as recited in claim [[4]] 3, further comprising, prior to receiving the request to upgrade the current trusted core:

generating the key;

having the key securely stored so that it can only be retrieved by the trusted core;

receiving requests to securely store data for applications executing on the computing device; and

using the key to encrypt the data.

8. **(Original)** A method as recited in claim 1, wherein the allowing comprises the new trusted core:

obtaining an encrypted key, wherein the key was previously used by the current trusted core to securely store data for applications executing on the same computing device as the current trusted core was executing on;

obtaining the key in decrypted form only if the new trusted core is the new trusted core that the trusted core intended to have access to the key;

generating a new key;

using the new key to retrieve data securely stored for applications after the upgrading; and

using the key to retrieve data securely stored for applications prior to the upgrading.

9. (Original) A method as recited in claim 1, wherein the allowing comprises the new trusted core obtaining a set of keys from a platform of the computing device, the set of keys including a key used by the current trusted core to securely store application data and a key to be used by the new trusted core to securely store application data.

10. (Currently amended) One or more computer readable media having stored thereon a plurality of instructions to facilitate upgrading a trusted core of an operating system installed on a computing device that, when executed by one or more processors, causes the one or more processors to:

identify a digest of a new trusted core to which the trusted core is to be upgraded; and

have a key securely stored, at least in part utilizing a binding key, so that [[it]] the key can only be retrieved by the new trusted core based on the digest of the new trusted core, wherein the key was previously used by the trusted core to securely store data for applications executing on the computing device.

11. (Original) One or more computer readable media as recited in claim 10, wherein the plurality of instructions further cause the one or more processors to, prior to having the key securely stored:

check whether the new trusted core satisfies one or more conditions; and

have the key securely stored only if the new trusted core satisfies the one or more conditions.

12. (Original) One or more computer readable media as recited in claim 11, wherein the one or more conditions comprise the new trusted core being a correct next version of the trusted core.

13. (Original) One or more computer readable media as recited in claim 11, wherein the one or more conditions comprise verifying that the digest of the new trusted core is digitally signed by the source of the new trusted core.

14. (Original) One or more computer readable media as recited in claim 10, wherein the having the key securely stored comprises invoking a seal operation that receives both the key and a cryptographic measure of the new trusted core, and that encrypts at least the key.

15. (Original) One or more computer readable media as recited in claim 14, wherein the cryptographic measure comprises a digest.

16. (Original) One or more computer readable media as recited in claim 10, wherein the instructions that cause the one or more processors to have the key securely stored so that it can only be retrieved by the new trusted core comprise instructions that cause the one or more processors to:

generate a temporary key;

encrypt, using the temporary key, a subset of the data securely stored for applications executing on the computing device; and

have the temporary key securely stored so that it can only be retrieved by the new trusted core, but not having the key securely stored so that it can be retrieved by the new trusted core.

17. (Original) One or more computer readable media as recited in claim 10, wherein the plurality of instructions further cause the one or more processors to, prior to identifying the digest:

generate the key;

have the key securely stored so that it can only be retrieved by the trusted core;

receive requests to securely store data for applications executing on the computing device; and

use the key to encrypt the data.

18. **(Original)** One or more computer readable media as recited in claim 17, wherein the instructions to generate the key comprises instructions to generate a random number to use as the key.

19. **(Original)** One or more computer readable media as recited in claim 17, wherein the instructions to generate the key comprises instructions to generate the key in a manner that allows the key to be calculated by the new trusted core but that does not allow the trusted core to calculate a new key generated by the new trusted core.

20. **(Currently amended)** One or more computer readable media having stored thereon a plurality of instructions to facilitate upgrading a trusted core of an operating system installed on a computing device to a new trusted core that, when executed by one or more processors, causes the one or more processors to:

obtain an encrypted key, wherein the key was previously used by the trusted core to securely store data for applications executing on the computing device;

obtain the key in decrypted form only if the new trusted core is the new trusted core that the trusted core intended to have access to the key;

generate a new key;

use the new key to retrieve data securely stored for applications after the upgrading; and

use the key to retrieve data securely stored for applications prior to the upgrading effective to enable both the trusted core and the new trusted core to subsequently retrieve the prior stored data while only the new trusted core can retrieve the after stored data.

21. (Original) One or more computer readable media as recited in claim 20, wherein the plurality of instructions further cause the one or more processors to:

identify a digest of another new trusted core to which the new trusted core is to be upgraded; and

have the new key securely stored so that it can only be retrieved by the other new trusted core based on the digest of the other new trusted core, wherein the new key is also used by the new trusted core to securely store data for applications executing on the computing device.

22. (Original) One or more computer readable media as recited in claim 20, wherein the key is a temporary key that was used by the trusted core to encrypt a subset of the data securely stored by the trusted core.

23. (Original) One or more computer readable media as recited in claim 20, wherein the decrypted key is obtained only if a digest of the new trusted core is the same as a digest that the trusted core identified as corresponding to a new trusted core that should have access to the key.

24. (Original) One or more computer readable media as recited in claim 20, wherein the plurality of instructions further cause the one or more processors to, after generating the new key, obtain the key previously used by the trusted core without having the encrypted key decrypted.

25. (Currently amended) A system comprising:

a processor;

a memory, coupled to the processor, configured to store an operating system with a trusted core and further configured to store a first plurality of instructions that, when executed by the processor, cause the processor to,

identify a digest of a new trusted core with which the trusted core is to be replaced, and

have a key securely stored, at least in part utilizing a binding key, so that ~~it~~ the key can only be retrieved by the new trusted core, wherein the key is previously used by the trusted core to securely store data for applications executing on the system, and

wherein the memory is further configured to store a second plurality of instructions that, when executed by the processor, cause the processor to,

obtain the decrypted key only if, based on the digest of the new trusted core, the new trusted core is the new trusted core that the trusted core intended to have access to the key;

generate a new key, and

use the new key to securely store and retrieve secrets for applications after the new trusted core replaces the trusted core.

26. (Original) A system as recited in claim 25, wherein the first plurality of instructions comprises the trusted core.

27. (Original) A system as recited in claim 25, wherein the second plurality of instructions comprises the new trusted core.

28. (Original) A system as recited in claim 25, wherein the first plurality of instructions further cause the processor to, prior to having the key securely stored:

check whether the new trusted core satisfies one or more conditions; and

have the key securely stored only if the new trusted core satisfies the one or more conditions.

29. (Original) A system as recited in claim 25, wherein the first plurality of instructions further cause the processor to:

generate a temporary key;

encrypt, using the temporary key, a subset of the data securely stored for applications executing on the system; and

have the temporary key securely stored so that it can only be retrieved by the new trusted core, but not having the key securely stored so that it can be retrieved by the new trusted core.

30. (Original) A system as recited in claim 25, wherein the first plurality of instructions further cause the processor to, prior to identifying the digest of the new trusted core and having the key securely stored:

generate the key;

have the key securely stored so that it can only be retrieved by the trusted core;

receive requests to securely store data for applications executing on the computing device; and

use the key to encrypt the data.

31. (Currently amended) A method comprising:

identifying a digest of a new trusted core to which a trusted core of an operating system installed on a computing device is to be upgraded; and

having a gatekeeper storage key encrypted, at least in part utilizing a binding key, so that ~~it~~ the key can only be retrieved by the new trusted core, wherein the gatekeeper storage key is a key previously used by the trusted core to securely store secrets for applications executing on the computing device.

32. (Original) A method as recited in claim 31, further comprising, prior to having the gatekeeper storage key encrypted:

checking whether the new trusted core satisfies one or more conditions; and

having the key securely stored only if the new trusted core satisfies the one or more conditions.

33. (Original) A method as recited in claim 31, wherein having a gatekeeper storage key encrypted so that it can only be retrieved by the new trusted core comprises:

generating a temporary key;

encrypting, using the temporary key, a subset of the data securely stored for applications executing on the computing device; and

having the temporary key securely stored so that it can only be retrieved by the new trusted core, but not having the key securely stored so that it can be retrieved by the new trusted core.

34. (Original) A method as recited in claim 31, further comprising, prior to identifying the digest of the new trusted core and having the gatekeeper storage key encrypted:

generating the key;

having the key securely stored so that it can only be retrieved by the trusted core;

receiving requests to securely store data for applications executing on the computing device; and

using the key to encrypt the data.

35. (Currently amended) A method comprising:

obtaining an encrypted gatekeeper storage key, wherein the gatekeeper storage key is a key previously used by a trusted core of an operating system installed on a computing device to securely store secrets for applications executing on the computing device;

obtaining the decrypted gatekeeper storage key only if, based on a digest of a new trusted core, the new trusted core is the new trusted core that the trusted core intended to have access to the gatekeeper storage key;

generating a new gatekeeper storage key;

using the new gatekeeper storage key to retrieve secrets securely stored for applications after the upgrading;

using the gatekeeper storage key to retrieve secrets securely stored for applications prior to upgrading to the new trusted core effective to enable both the trusted core and the new trusted core to subsequently retrieve the prior stored data while only the new trusted core can retrieve the after stored data.

36. (Original) A method as recited in claim 35, further comprising:

identifying a digest of another new trusted core to which the new trusted core is to be upgraded; and

having the new gatekeeper storage key securely stored so that the gatekeeper storage key can only be retrieved by the other new trusted core based on the digest of the other new trusted core, wherein the new gatekeeper storage key is also used by the new trusted core to securely store data for applications executing on the computing device.

37. (Original) A method as recited in claim 35, wherein the gatekeeper storage key is a temporary gatekeeper storage key that was used by the trusted core to encrypt a subset of the data securely stored by the trusted core.

38. (Original) A method as recited in claim 35, wherein the decrypted gatekeeper storage key is obtained only if a digest of the new trusted core is the same as a digest that the trusted core identified as corresponding to a new trusted core that should have access to the gatekeeper storage key.

39. (Currently amended) One or more computer readable media having stored thereon a plurality of instructions to facilitate upgrading a trusted core of an operating system installed on a computing device to a new trusted core that, when executed by one or more processors, causes the one or more processors to:

obtain a set of one or more keys, wherein a first key of the set of keys was previously used by the trusted core to securely store data for applications executing on the computing device;

use a second key of the set of keys to securely store data for applications after the upgrading;

use the first key to retrieve data securely stored for applications prior to the upgrading; and

use the second key to retrieve data securely stored for applications after the upgrading effective to enable both the trusted core and the new trusted core to subsequently retrieve the prior stored data while only the new trusted core can retrieve the after stored data.

40. (Original) One or more computer readable media as recited in claim 39, wherein the set of one or more keys includes a plurality of additional keys, and wherein each of the plurality of additional keys was used by a different trusted core prior to upgrading to the trusted core.

41. (Original) One or more computer readable media as recited in claim 39, wherein a value SK_n refers to the set of one or more keys, an operation SHA-1 refers to the Secure Hash Algorithm 1, an operation cat refers to a concatenation operation, a value *BK* refers to a platform key, a value *public_key* refers to a public key of a party that generated the new trusted core, a value *n* refers to a particular trusted core number, a value *N* refers to a number of the trusted core, and wherein the plurality of instructions to obtain the set of one or more keys comprises instructions that cause the one or more processors to have the set of one or more keys generated as follows:

SK_n=SHA-1(cat(*BK*, *public_key*, *n*), for n=0 to N.

42. (Currently amended) A method comprising:

requesting, by a trusted core, a set of keys to be used by the trusted core for secure secret storage and retrieval on a computing device, wherein the set of keys includes a first key and a second key, and wherein the first key was previously used by a previous trusted core for secure secret storage prior to the computing device being upgraded to the trusted core;

using the second key for secure storage of secrets by applications by the trusted core;

using the second key for retrieval of secrets previously stored by applications via the trusted core;

using the first key for retrieval of secrets previously stored by applications via the previous trusted core effective to enable both the trusted core and the previous trusted core to subsequently retrieve the secrets previously stored by applications via the previous trusted core while only the trusted core can retrieve secrets secured by the second key.

43. (Original) A method as recited in claim 42, wherein a value SK_n refers to the set of keys, an operation SHA-1 refers to the Secure Hash Algorithm 1, an operation cat refers to a concatenation operation, a value *BK* refers to a key of a platform of the computing device, a value *public_key* refers to a public key of a party that generated the trusted core, a value *n* refers to a particular trusted core number, a value *N* refers to a number of the trusted core, and wherein the plurality of instructions to request the set of keys comprises instructions that cause the one or more processors to have the set of one or more keys generated as follows:

SK_n=SHA-1(cat(*BK*, *public_key*, *n*), for n=0 to N.